高速多項式GCD計算法(数式処理における理論とその応用の研究)

AUTHOR(S):

元吉, 文男

CITATION:

元吉, 文男. 高速多項式GCD計算法(数式処理における理論とその応用の研究). 数理解析研究所講究録 1996, 941: 151-155

ISSUE DATE:

1996-03

URL:

http://hdl.handle.net/2433/60125

数理解析研究所講究録
941 巻 1996 年 151-155

19.

高速多項式GCD計算法

元吉文男 (電総研)

19.1 はじめに

多項式に対して拡張 Euclid 法の計算をする際に、高速乗算アルゴリズムを使用すると計算量のオーダが改善されることを示す。

19.2 拡張Euclid法

まず、通常の拡張 Euclid 法を分析する。このアルゴリズムは以下のように記述される。

拡張 Euclid アルゴリズム $E(G,H)$

入力： G,H　　一変数多項式

出力： $gcd(G,H),A,B$　　s.t. $AG+BH=gcd(G,H)$

$(u_0,v_0,w_0)=(1,0,G)$

$(u_1,v_1,w_1)=(0,1,H)$

for $i=1,2,\cdots$ **until** $w_i=0$

$q_i=\lfloor w_{i-1}/w_i \rfloor$

$(u_{i+1},v_{i+1},w_{i+1})=(u_{i-1},v_{i-1},w_{i-1})-q_i(u_i,v_i,w_i)$

$(A,B,gcd(G,H))=(u_{i-1},v_{i-1},w_{i-1})$

以上のアルゴリズムの計算量を $\deg A = n$, $\deg B = n-1$ の場合について考察する。

計算時間が最もかかるのは $\deg w_i = \deg w_{i-1} - 1$、かつ GCD が 1 となるときであるので、この場合が最悪の計算量となる。上記アルゴリズムを分析すると $\deg w_i = n - i$, $\deg u_i = i - 2$, $\deg v_i = i - 1$ となることがわかる。このとき $\deg q_i = 1$ となっている。q_i、w_{i+1}は除算を普通に実行して同時に計算でき、u_{i+1}、u_{i+1}の計算は 1 次式を掛けてから加える操作なので、結局以下の表に示す演算回数になる。

各ステップでの演算回数

	除算	乗算	加減算
q_i, w_{i+1}	2	$2(n-i)$	$2(n-i)$
u_{i+1}		$2(i-1)$	$2(i-2)$
v_{i+1}		$2i$	$2(i-1)$
合計	$2i$	$2n+2i-2$	$2n+2i-6$

全体の演算回数

除算	乗算	加減算
$2n$	$n(3n-1)$	$n(3n-5)$

$E(G,H)$ の分析

元の多項式の低次の項はある時 $(i=k)$ まで q_i の値に影響を与えない。実際、j次未満の項は w_kの $k+j$次までしか影響せず、また w_kは $n-k$次であることから、$n-k > k+j$すなわち $2k < n-j$ならば、j次未満の項を無視しても q_iの値に影響を与えない。そこで、ある回数までは j次より高次の項について計算を行ない、そこで無視しておいた項に u_k, v_k 等を掛ければ、その時点での正しい w_kが得られる。このとき乗算に高速乗算算法を使用することによって全体の高速化が行なえるはずである。

19.3 高速拡張 Euclid 算法

前節の考察したように、次数に対してある割合までは高速に部分剰余が計算できそうである。そこで次のような手続きを考え、それを繰り返し適用することによって GCD を計算することにする。その手続き $P(U)$ のアルゴリズムと図式化を次頁以下に記す。なお、w_k, u_k, v_k などは $E(G,H)$ と同じ定義とする。また、実際にはこの算法は大きな次数について効果的であるので、ある次数以下のときは $E(G,H)$ を使用するような判断が $P(U)$ に組込まれることになる。

$k = \lfloor (1-\alpha)n \rfloor$, $(0 < \alpha < 1)$ とする。

高速拡張 Euclid アルゴリズム $P(U)$

入力 :$U = \begin{pmatrix} G \\ H \end{pmatrix}$, $(\deg G = n, \deg H = n-1)$

出力 :$V = \begin{pmatrix} w_k \\ w_{k+1} \end{pmatrix}, W = \begin{pmatrix} u_k & v_k \\ u_{k+1} & v_{k+1} \end{pmatrix}$ $s.t.\ V = WU$

$U = U_0 x^{(1-\beta)n} + U_1$ $(\deg U_1 = (1-\beta)n - 1)$ と分解する

$(X_1, W_1) = P(U_0)$

$(X_2, W_2) = P(X_1)$

$(V, W) = (X_2 x^{(1-\beta)n} + W_2 W_1 U_1,\ W_2 W_1)$

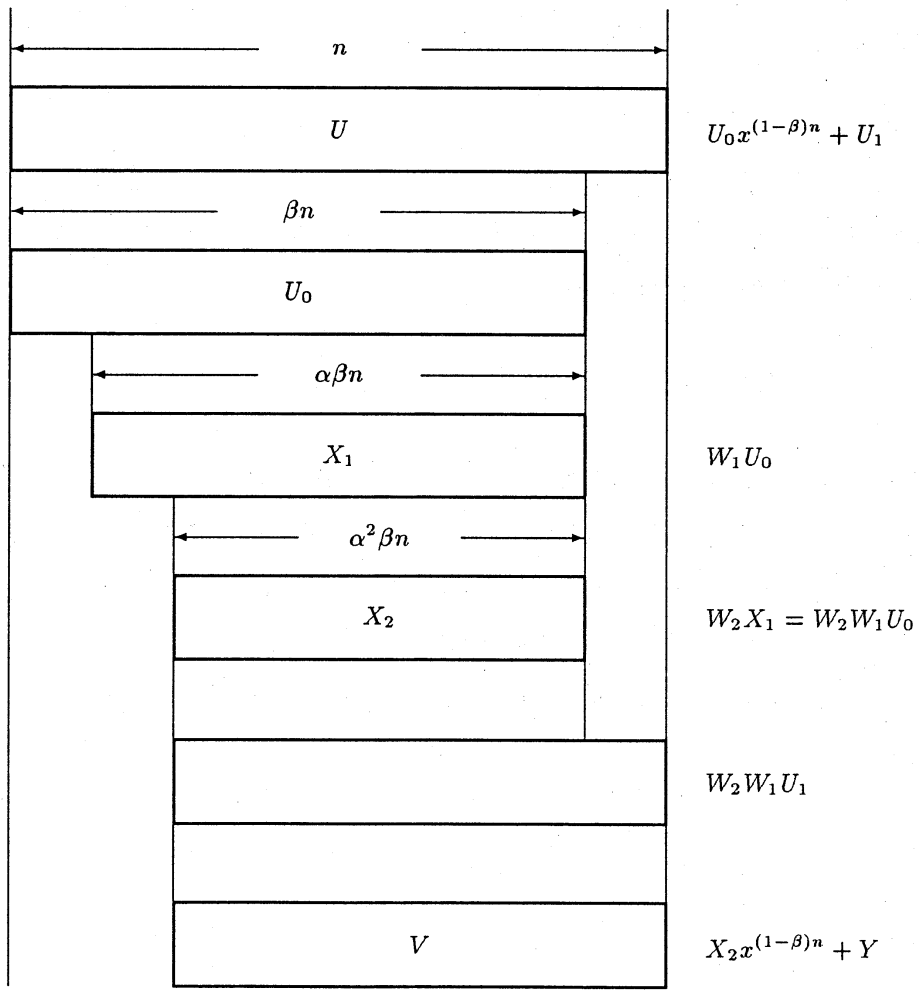

図 1　高速拡張ユークリッド算法の図式化

以上のアルゴリズムで α と β は任意に決められるわけではなく次のような条件が必要である。すなわち、$P(U)$ を再帰的に適用するために V の次数が αn より小さい必要がある。$\deg V = \deg X_2 + \deg U_1 + 1$ であるから第一の条件として

$$\alpha \geq \alpha^2\beta + (1-\beta)$$

が得られる。次に $W_2W_1U_1$ の次数が V の次数を越えてはいけないこと、および $\deg W_2W_1 = (\beta - \alpha^2\beta)n$ であることから第二の条件として

$$(1-\beta) + (\beta - \alpha^2\beta) \geq \alpha$$

が得られる。また、なるべく高速に計算を実現するには β をなるべく小さくしたい。上の 2 つの条件の下でβを最小にするには

$$\alpha = \frac{\sqrt{2}}{2}, \quad \beta = \frac{1}{1+\alpha}$$

とすればよいことがわかる。

$P(U)$ の演算時間

$\deg U = n$ のときのアルゴリズム $P(U)$ の演算時間を $N(n)$ とする。2 回の P の適用では $N(\beta n) + N(\alpha\beta n)$ の時間が、W_2W_1 の計算には $8M((1-\alpha)\beta n)$ の時間が、$(W_2W_1)U_1$の計算には $4M((1-\alpha^2)\beta n)$ の時間がかかるので最終的には

$$N(n) = N(\beta n) + N(\alpha\beta n) + 8M((1-\alpha)\beta n) + 4M((1-\alpha^2)\beta n)$$

となる。

ここで、n 次の多項式同士の乗算時間が $M(n) = dn^\varsigma$ の算法を使用するとする。$N(n) = c(n)n^\gamma$, ($\lim_{n\to\inf} c(n)/x^\delta = 0, \quad \delta > 0$) とすると $\lim M(n)/N(n) = 0$ の場合はγの解が存在しない (上記のα, β の下で)。そこで$\gamma = \varsigma$の場合を調べる。

$$cn^\varsigma = c\beta^\varsigma n^\varsigma + c\alpha^\varsigma\beta^\varsigma n^\varsigma + 8d(1-\alpha)^\varsigma\beta^\varsigma n^\varsigma + 4d(1-\alpha^2)^\varsigma\beta^\varsigma n^\varsigma$$

$$c = c\beta^\varsigma + c\alpha^\varsigma\beta^\varsigma + 8d(1-\alpha)^\varsigma\beta^\varsigma + 4d(1-\alpha^2)^\varsigma\beta^\varsigma$$

となるので

$$c = 4d\,\frac{2(1-\alpha)^\varsigma\beta^\varsigma + (1-\alpha^2)^\varsigma\beta^\varsigma}{1-\beta^\varsigma - \alpha^\varsigma\beta^\varsigma}$$

となる。

アルゴリズム $P(U)$ は部分剰余の計算であり、拡張 Euclid 算法の最後まで計算するには、各 W_iの

積を計算する部分も含めると演算時間 $Q(n)$ は

$$\begin{aligned} Q(n) &= N(n) + N(\alpha n) + N(\alpha^2 n) + \cdots + 8(M(\alpha n) + M(\alpha^2 n) + \cdots) \\ &= c(1 + \alpha^\varsigma + \alpha^{2\varsigma} + \cdots)n^\varsigma + 8d(1 + \alpha^\varsigma + \alpha^{2\varsigma} + \cdots)n^\varsigma \\ &= \frac{c + 8d}{1 - \alpha^\varsigma} n^\varsigma \end{aligned}$$

の時間が必要になる。

$\varsigma = \log_2 3$ およびα, βの数値を代入すると $N(n) \simeq 24.5n^{1.58}$, $Q(n) \simeq 200.0n^{1.58}$になる。これが普通の方法の計算時間 $6n^2$ より早くなるのは

$$n \simeq 4700$$

のときである。

19.4 おわりに

拡張 Euclid 算法を計算量のオーダから見て高速に実行する方法について考察し、多項式の乗算として次数のべき乗時間で計算できる方法を使用したときには、オーダ的には乗算と同じ時間で計算できることを示した。たが、特に大きな次数の多項式の必要がないかぎり、上に結果のように実際的な観点からは採用する効果が薄いと思われる。もちろん、普通の方法と高速算法とを混在させて使用することによって前節の見積もりよりは小さな次数について速度が改善されるが、それでも実際に必要になる次数では、通常の方法で十分であろう。